Optimization of Mechanical Design Problems Using Improved Differential Evolution Algorithm

Millie Pant, Radha Thangaraj and V. P. Singh
Department of Paper Technology,
Indian Institute of Technology Roorkee, India.
millifpt@iitr.ernet.in, t.radha@ieee.org, singhfpt@iitr.ernet.in

***Abstract*— Differential Evolution (DE) is a novel evolutionary approach capable of handling non-differentiable, non-linear and multi-modal objective functions. DE has been consistently ranked as one of the best search algorithm for solving global optimization problems in several case studies. This paper presents an Improved Constraint Differential Evolution (ICDE) algorithm for solving constrained optimization problems. The proposed ICDE algorithm differs from unconstrained DE algorithm only in the place of initialization, selection of particles to the next generation and sorting the final results. Also we implemented the new idea to five versions of DE algorithm. The performance of ICDE algorithm is validated on four mechanical engineering problems. The experimental results show that the performance of ICDE algorithm in terms of final objective function value, number of function evaluations and convergence time.**

***Index Terms*—Differential Evolution, optimization, Mechanical design problems, constraint optimization.**

I. Introduction

Many real-world optimization problems are solved subject to sets of constraints. The search space in COPs consists of two kinds of solutions: feasible and infeasible. Feasible points satisfy all the constraints, while infeasible points violate at least one of them. Therefore the final solution of an optimization problem must satisfy all constraints. A constrained optimization problem may be distinguished as a Linear Programming Problem (LPP) and Nonlinear Programming Problem (NLP). In this paper we have considered NLP problems where either the objective function or the constraints or both are nonlinear in nature. The general NLP is given by nonlinear objective function f, which is to be minimized/maximized with respect to the design variables $\overline{x} = (x_1, x_2, ..., x_n)$ and the nonlinear inequality and equality constraints. This can be formulated by,

$$Minimize\ /\ Maximize\quad f(\overline{x})$$

$$Subject\ \ to: \quad g_j(\overline{x}) \le 0, \quad j = 1, ..., p \qquad (1)$$

$$h_k(\overline{x}) = 0, \quad k = 1, ..., q \qquad (2)$$

$$x_{i\min} \le x_i \le x_{i\max} \ (i = 1, ..., n).$$

where p and q are the number of inequality and equality constraints respectively. There are many traditional methods in the literature for solving NLP. However, most of the traditional methods require certain auxiliary properties (like convexity, continuity etc.) of the problem and also most of the traditional techniques are suitable for only a particular type of problem (for example Quadratic Programming Problems, Geometric Programming Problems etc). Keeping in view the limitations of traditional techniques researchers have proposed the use of stochastic optimization methods and intelligent algorithms for solving NLP which may be constrained or unconstrained. Some examples are: Genetic Algorithms [1] – [3], Ant Colony Optimization [4], Chaos Optimization Algorithm [5], Particle Swarm Optimization [6], Differential Evolution [7] etcetera. Based on the research efforts in literature, constraint handling methods have been categorized in a number of classes [8] **-** [10]:

- Reject infeasible solutions
- Penalty function methods
- Convert the constrained problem to an unconstrained problem
- Preserving feasibility methods
- Pareto ranking methods
- Repair methods

In the present research paper we have concentrated our work to DE, which is comparatively a newer addition to the class of population based search techniques. DE is a stochastic, population based search strategy developed by Storn and Price [7] in 1995. It is a novel evolutionary approach capable of handling no-differentiable, non-linear and multimodal objective functions. DE has been designed as a stochastic parallel direct search method, which utilizes concepts borrowed from the broad class of EAs. The method typically requires few, easily chosen control parameters. This paper presents an Improved Constraint Differential Evolution (ICDE) algorithm for solving constrained optimization problems. The structure of the paper is as follows: in section II, we have briefly explained the Differential Evolution Algorithm, in section III; we have defined and explained the proposed ICDE algorithm. Section IV deals with experimental settings and test problems, Section V gives the numerical results and discussion and finally the paper conclude with section VI.

II. Differential Evolution Algorithm

DE shares a common terminology of selection, crossover and mutation operators with GA however it is the application of these operators that make DE different from GA. Whereas, in GA crossover plays a significant role, it is the mutation operator which effects the working of DE [11].

The working of DE may be described as follows:

***Mutation*:** For a D-dimensional search space, for each target vector $X_{i,g}$ at the generation g, its associated

DOI: 01.IJPIE.02.01.205

mutant vector is generated via certain mutation strategy. The most frequently used mutation strategies implemented in the DE codes are listed below.

DE/rand/1(S1): $V_{i,g} = X_{r_1,g} + F*(X_{r_2,g} - X_{r_3,g})$ (3)

DE/rand/2 (S2):

$$V_{i,g} = X_{r_1,g} + F*(X_{r_2,g} - X_{r_3,g}) + F*(X_{r_4,g} - X_{r_5,g}) \quad (4)$$

DE/bes t/1 (S3): $V_{i,g} = X_{best,g} + F*(X_{r_1,g} - X_{r_2,g})$ (5)

DE/best/2 (S4):

$$V_{i,g} = X_{best,g} + F*(X_{r_1,g} - X_{r_2,g}) + F*(X_{r_3,g} - X_{r_4,g}) \quad (6)$$

DE/rand-to-best/1 (S5):

$$V_{i,g} = X_{r_1,g} + F*(X_{best,g} - X_{r_2,g}) + F*(X_{r_3,g} - X_{r_4,g}) \quad (7)$$

where $r_1, r_2, r_3, r_4, r_5 \in \{1,2,....,NP\}$ are randomly chosen integers, must be different from each other and also different from the running index i. F (>0) is a scaling factor which controls the amplification of the difference vector. $X_{best,g}$ is the best individual vector with the best fitness value in the population at generation g.

***Crossover*:** In order to increase the diversity of the perturbed parameter vectors, crossover is introduced [12]. The parent vector is mixed with the mutated vector to produce a trial vector $u_{ji,g+1}$

$$u_{ji,g+1} = \begin{cases} v_{ji,g+1} & if \ (rand_j \le CR) \ or \ \ (j = j_{rand}) \\ x_{ji,g} & if \ (rand_j > CR) \ and \ (j \ne j_{rand}) \end{cases} \quad (8)$$

where j = 1, 2,……, D; $rand_j \in [0,1]$; CR is the crossover constant takes values in the range [0, 1] and $j_{rand} \in (1,2,....., D)$ is the randomly chosen index.

***Selection*:** Selection is the step to choose the vector between the target vector and the trial vector with the aim of creating an individual for the next generation. The simple flow of DE algorithm is given in Fig 1.

```
Initialize the population
Calculate the fitness value for each particle
Do
For i = 1 to number of particles
        Do mutation, Crossover and
Selection
End for.
Until stopping criteria is reached.
```

Fig 1 Flow of DE algorithm

III. Proposed ICDE Algorithm

The proposed algorithm ICDE is a simple algorithm for solving constraint optimization problems, it is easy to implement. It differs from unconstrained optimization algorithm only in the place of initialization, selection of particles to the next generation and sorting the final results. The proposed ICDE algorithm uses the mean zero standard deviation one normal distribution for initializing the population and uses the following three selection criteria: After calculating the trial vector (i) If the trial vector and the target vector are feasible then select the best one (ii) If both the particles are infeasible then select the one having smaller constraint violation (iii) If one is feasible and the other one is infeasible then select the feasible one. Also at the end of every iteration, the particles are sorted by using the three criteria: (a) Sort feasible solutions in front of infeasible solutions (b) Sort feasible solutions according to their fitness function values (c) Sort infeasible solutions according to their constraint violations. The computational steps of ICDE algorithm is given below:

```
Step 1 Initialize the population using
normal distribution with mean zero and
standard deviation one.
Step 2 For all particles
Evaluate the objective function
Calculate the constraint violation
End for
Step 3 While stopping criterion is not
satisfied
Do
Step 3.1 Mutation
For all particles
Generate a mutated vector V_i,g
corresponding to the target vector X_i,g
via one of the equations (3) to (7)
End for
Step 3.2 Crossover //Generate trial
                         vector U_i,g
For all particles
Select j_rand∈ {1,…,D}
For j = 1 to D
If (rand(0,1) d” CR or j= j_rand)
Then U_i,g = V_i,g
Else U_i,g = X_i,g
End if
End for
End for
Step 3.3 Selection
For all particles
Set X_i,g+1 according to the three selection
criteria
End for
Step 3.4 Sort the particles using the
           three sorting rules
Step 3.5 Go to next generation
Step 4 End while
```

IV. Experimental Settings and Test Problems

In order to make a fair comparison of all versions of DE algorithms, we fixed the same seed for random number generation so that the initial population is same for both the algorithms. The population size is taken as 50. The crossover constant CR is set as 0.95 and the scaling factor F is set as

DOI: 01.IJPIE.02.01.205

0.7. For each algorithm, the stopping criteria is to terminate the search process when one of the following conditions is satisfied: (1) the maximum number of generations is reached (assumed 10000 generations), (2) $| f_{max} - f_{min} | < 10^{-4}$ where *f* is the value of objective function. A total of 30 runs for each experimental setting were conducted. If the run satisfies the second stopping condition then that run is called successful run. Also we implemented the new idea to five versions of DE algorithm. To check the efficiency of the proposed ICDE algorithm we have tested it on four optimization problems arising commonly in the field of Mechanical engineering. All the problems considered here are highly non linear in nature and are subject to various constraints. The mathematical models of these problems may be given as:

A. Weight Minimization of a Speed Reducer (WMSR) [13]

The mathematical model of this problem is,

Min

$$f(x) = 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934)$$

$$-1.508 x_1 (x_6^2 + x_7^2) + 7.477 (x_6^3 + x_7^3) + 0.7854 (x_4 x_6^2 + x_5 x_7^2)$$

Subject to

$$x_1 x_2^2 x_3 \geq 27\,,\ x_1 x_2^2 x_3^2 \geq 397.5\,,\ x_2 x_6^4 x_3 x_4^{-3} \geq 1.93\,,$$

$$A_1 B_1^{-1} \leq 1100$$

Where $A_1 = [(745 x_4 x_2^{-1} x_3^{-1})^2 + 16.9611^6]^{0.5}$, $B_1 = 0.1 x_6^3$

$$A_2 B_2^{-1} \leq 850$$

Where $A_2 = [(745 x_5 x_2^{-1} x_3^{-1})^2 + 15.7510^6]^{0.5}$, $B_2 = 0.1 x_7^3$

$$x_2 x_3 \leq 40\,,\ x_1 x_2^{-1} \geq 5\,,\ x_1 x_2^{-1} \leq 12\,,\ 1.5 x_6 - x_4 \leq -1.9\,,\ 1.5 x_7 - x_5 \leq -1.9 .$$

$$2.6 \leq x_1 \leq 3.6\,,\quad 0.7 \leq x_2 \leq 0.8\,,\quad 17 \leq x_3 \leq 28\,,$$

$$7.3 \leq x_4 \leq 8.3\,,\ 7.3 \leq x_5 \leq 8.3\,,\ 2.9 \leq x_6 \leq 3.9\,,\ 5 \leq x_7 \leq 5.5$$

B. Heat Exchanger Network Design (HEND) [14]

The mathematical model is,

Minimize $f(x) = x_1 + x_2 + x_3$

Subject to

$$-1 + 0.0025(x_4 + x_6) \leq 0\,,\ -1 + 0.0025(x_5 + x_7 - x_4) \leq 0\,,$$

$$-1 + 0.01(x_8 - x_5) \leq 0 \quad ,$$

$$-x_1 x_6 + 833.33252 x_4 + 100 x1 - 83333.333 \leq 0\,,$$

$$-x_2 x_7 + 1250 x_5 + x_2 x_4 - 1250 x_4 \leq 0\,,$$

$$-x_3 x_8 + 1250000 + x_3 x_5 - 2500 x_5 \leq 0\,,$$

$$-100 \leq x_1 \leq 10000\,,\ 1000 \leq x_i \leq 10000\ (i = 2,3)\,,$$

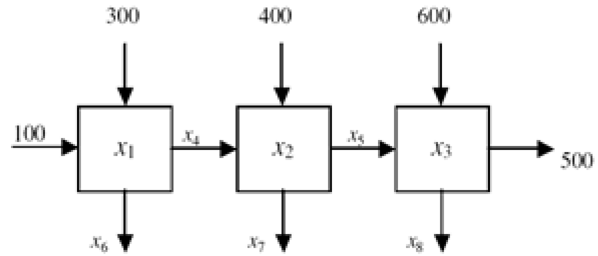

Fig 2 Heat Exchanger Network Design Problem

C. Gas Transmission Compressor Design (GTCD) [15]

The mathematical model is,

$$f(x) = 8.61 \times 10^5 x_1^{1/2} x_2 x_3^{-2/3} x_4^{-1/2} + 3.69 \times 10^4 x_3$$

$$+ 7.72 \times 10^8 x_1^{-1} x_2^{0.219} - 765.43 \times 10^6 x_1^{-1}$$

Subject to

$$x_4 x_2^{-2} + x_2^{-2} \leq 1$$

$$20 \leq x_1 \leq 50\,,\ 1 \leq x_2 \leq 10\,,\ 20 \leq x_3 \leq 50\,,\ 0.1 \leq x_4 \leq 60$$

D. Optimal Design of Industrial refrigeration System (ODIRS) [16]

The mathematical model is,

$$f(x) = 63098.88 x_2 x_4 x_{12} + 5441.5 x_2^2 x_{12} + 115055.5 x_2^{1.664} x_6$$

$$+ 6172.27 x_2^2 x_6 + 63098.88 x_1 x_3 x_{11} + 5441.5 x_1^2 x_{11}$$

$$+ 115055.5 x_1^{1.664} x_5 + 6172.27 x_1^2 x_5 + 140.53 x_1 x_{11}$$

$$+ 281.29 x_3 x_{11} + 70.26 x1^2 + 281.29 x_1 x_3 + 281.29 x_3^2$$

$$+ 14437 x_8^{1.8812} x_{12}^{0.3424} x_{10} x_{14}^{-1} x_1^2 x_7 x_9^{-1}$$

$$+ 20470.2 x_7^{2.893} x_{11}^{0.316} x_1^2$$

Subject to

$$1.524 x_7^{-1} \leq 1\,,\ 1.524 x_8^{-1} \leq 1\,,\ 0.07789 x_1 - 2 x_7^{-1} x_9 \leq 1\,,$$

$$7.05305 x_9^{-1} x_1^2 x_{10} x_8^{-1} x_2^{-1} x_{14}^{-1} \leq 1\,,\ 0.0833 x_{13}^{-1} x_{14} \leq 1\,,$$

$$47.136 x_2^{0.333} x_{10}^{-1} x_{12} - 1.333 x_8 x_{13}^{2.1195}$$

$$+ 62.08 x_{13}^{2.1195} x_{12}^{-1} x_8^{0.2} x_{10}^{-1} \leq 1\,,$$

$$0.04771 x_{10} x_8^{1.8812} x_{12}^{0.3424} \leq 1\,,$$

$$0.0488 x_9 x_7^{1.893} x_{11}^{0.316} \leq 1\,,\qquad 0.0099 x_1 x_3^{-1} \leq 1\,,$$

$$0.0193 x_2 x_4^{-1} \leq 1\,,\quad 0.0298 x_1 x_5^{-1} \leq 1\,,\quad 0.056 x_2 x_6^{-1} \leq 1\,,$$

$$2 x_9^{-1} \leq 1\,,\ 2 x_{10}^{-1} \leq 1\,,\ x_{12} x_{11}^{-1} \leq 1\,,$$

$$0.001 \leq x_i \leq 5\,,\ i = 1,\ldots,14$$

V. Experimental Results and Discussion

Tables I – IV gives the numerical results given four real life constrained optimization problems. These problems occur frequently in the field of mechanical designs. The comparison criteria for the algorithms is done in terms of best, average and worst fitness function values, NFE, std, SR and time. For

DOI: 01.IJPIE.02.01.205

all the algorithms, NFE represents the number of function evaluations, which helps in determining the convergence of the algorithm. Lesser value of NFE implies faster convergence. std represents the standard deviation which tells the stability of the algorithms. Smaller std implies that the algorithm is more stable. SR represents the success rate, which signifies the efficiency of an algorithm. It tells us how many times the algorithm was able to converge successfully within 1% of the true global optima. For all the problems we compared our results with those available in literature. Form Tables I to IV, the results obtained by the different DE versions and the ones available in literature are given. From the numerical results it is quite visible that the versions of DE gave better results than the ones available in literature. In terms of best, average and worst fitness function values all the algorithms gave more or less similar results. However in terms of NFE, SR and time taken, the algorithms showed different behavior.

TABLE I COMPARISON RESULTS OF WMSR

Item	DE/rand/1	DE/rand/2	DE/best/1	DE/best/2	DE/rand-to-best/1	Result in [13]
Best	**2863.36**	**2863.36**	**2863.36**	**2863.36**	**2863.36**	2994.47
Average	2863.36	2863.36	2875.28	2866.52	2881.15	-NA-
Worst	2863.36	2863.36	3022.43	2902.89	2922.53	-NA-
Std	**1.56e-05**	1.84e-05	32.1611	10.7244	20.1398	-NA-
NFE	9112	19232	3802	7458	**3672**	-NA-
SR	**100%**	**100%**	**100%**	**100%**	**100%**	-NA-
Time (sec)	1.44	3.16	0.52	1.08	**0.48**	-NA-

TABLE II COMPARISON RESULTS OF HEND

Item	DE/rand/1	DE/rand/2	DE/best/1	DE/best/2	DE/rand-to-best/1	Result in [14]
Best	7049.25	7049.25	7049.25	7049.25	7049.25	7049.25
Average	7049.25	7049.25	7067.24	7049.25	7049.25	-NA-
Worst	7049.25	7049.25	7373.79	7049.25	7049.25	-NA-
Std	6.17e-05	3.33e-05	63.401	**1.25e-05**	1.45e-05	-NA-
NFE	86316	381338	67598	194826	277594	**33146**
SR	**100%**	**100%**	96%	**100%**	**100%**	88%
Time (sec)	**0.16**	0.76	**0.16**	0.4	0.48	1.292

TABLE III COMPARISON RESULTS OF GTCD

Item	DE/rand/1	DE/rand/2	DE/best/1	DE/best/2	DE/rand-to-best/1	Result in [15]
Best	**2.963e+06**	**2.963e+06**	**2.963e+06**	**2.963e+06**	**2.963e+06**	2.99e+06
Average	2.963e+06	2.963e+06	2.963e+06	2.963e+06	2.963e+06	-NA-
Worst	2.963e+06	2.963e+06	2.963e+06	2.963e+06	2.963e+06	-NA-
Std	8.79e-06	7.99 e-06	**3.17 e-06**	4.85 e-06	6.08 e-06	-NA-
NFE	14634	28430	**6640**	18610	23284	-NA-
SR	**100%**	**100%**	**100%**	**100%**	**100%**	-NA-
Time (sec)	0.56	1.0	**0.28**	0.64	0.84	-NA-

TABLE IV COMPARISON RESULTS OF ODIRS

Item	DE/rand/1	DE/rand/2	DE/best/1	DE/best/2	DE/rand-to-best/1	Result in [16]
Best	**13646.5**	27275.6	13646.6	13651.9	14399.6	19230
Average	13646.5	34972	14282.5	13660.8	15336.5	-NA-
Worst	13646.5	40908.8	17323.3	13683	18166.6	-NA-
Std	**7.82e-05**	5557.24	1118.68	10.4135	777.945	-NA-
NFE	316526	500050	**185504**	500050	500050	-NA-
SR	**100%**	0%	96%	0%	0%	-NA-
Time (sec)	92.72	150.64	**56.16**	220.04	157.08	-NA-

DOI: 01.IJPIE.02.01.205

The first problem is involves the design of a speed reducer for small aircraft engine. It has a nonlinear objective function and it consists of eleven inequality constraints and seven unknown variables. For this problem all the DE versions gave same results in terms of best, worst and average fitness function values. If we compare the NFC and convergence time then DE/rand-to-best/1 is better than all other versions. The second problem addresses the design of a heat exchanger network as shown in Fig 2. It has three equality constraints, three inequality constraints and eight decision variables. For this problem also all the algorithms gave same result in comparison best fitness function value. In comparison of average fitness value DE/best/1 gave slightly worse value than other algorithms. But in terms of convergence time it is better than all other versions. The third problem is a gas transmission compressor design problem. For this problem DE/best/1 gave better result in terms of standard deviation, NFE and convergence time. DE/rand/1 gave better result than other algorithms in terms of best fitness function value.

VI Conclusion

In this paper, we proposed an Improved DE algorithm called ICDE for solving constrained optimization problems. ICDE differs from the basic DE in the initialization, selection and sorting phases. These modifications are embedded on various versions of DE and their efficiency is validated on a set of four real life engineering design problems, occurring frequently in the field of mechanical engineering. We would like to add that in the present article though we have considered only four problems, the preliminary numerical results obtained, show that the proposed modifications are beneficial for solving constrained optimization problems effectively. Moreover, this is a general technique/ modification and can be applied to any population based search technique like Genetic Algorithms, Particle Swarm Optimization etc.

References

[1] Y. Li, M.Gen, "Non-linear mixed integer programming problems using genetic algorithm and penalty function", In Proc. of IEEE Int. Conf. on SMC, pp. 2677 – 2682, 1996.

[2] Y. Takao, M. Gen, T. Takeaki, Y. Li, "A method for interval 0-1 number non-linear programming problems using genetic algorithm", Computers and Industrial Engineering, Vol. 29, pp. 531 – 535, 1995.

[3] J. F. Tang, D.Wang, et al., "A hybrid genetic algorithm for a type of nonlinear programming problem", Computer Math. Applic, Vol. 36(5), pp. 11 – 21, 1998.

[4] M. Dorigo, V. Maniezzo, A. Colori, "Ant system optimization by a colony of cooperating agents", IEEE Trans. on system, Man, and Cybernetics, Vol. 26(1), pp. 28 – 41, 1996.

[5] Z. L. Wang, L. Qiu, L. Function, C. Liang, "Application of chaos optimization algorithm to nonlinear constrained programming", Journal of North China Institute of Water Conservancy and Hydroelectric Power, Vol. 23(2), pp. 1 -3, 2002.

[6] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV, pp. 1942-1948, 1995.

[7] R. Storn and K. Price, "Differential Evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces", Technical Report, International Computer Science Institute, Berkley, 1995.

[8] G. Coath, S. K. Halgamuge, "A Comparison of Constraint-Handling Methods for the Application of Particle Swarm Optimization to Constrained Nonlinear Optimization Problems", In Proc. of the IEEE Congress on Evolutionary Computation, Vol. 4, pp. 2419 – 2425, 2003.

[9] S. Koziel, Z. Michalewicz, "Evolutionary Algorithms, Homomorphus Mappings, and Constrained Optimization", Evolutionary Computation, Vol. 7(1), pp. 19 – 44, 1999.

[10] Z. Michalewicz, "A Survey of Constraint Handling Techniques in Evolutionary Computation Methods", In Proc. of the Fourth Annual Conf. on Evolutionary Programming, pp. 135 – 155, 1995.

[11] D. Karaboga and S. Okdem, "A simple and Global Optimization Algorithm for Engineering Problems: Differential Evolution Algorithm", Turk J. Elec. Engin. 12(1), 2004, pp. 53 – 60.

[12] R. Storn and K. Price, "Differential Evolution – a simple and efficient Heuristic for global optimization over continuous spaces", Journal Global Optimization. 11, 1997, pp. 341 – 359.

[13] Floudas, C.A., Pardalos, P.M., "A collection of test problems for constrained global optimization algorithms", LNCS. Springer, Germany, 1990.

[14] B. V. Babu and R. Angira, "Optimization of Industrial Processes Using Improved and Modified Differential Evolution", Studies in Fuzziness and Soft Computing, Vol. 226, 2008, pp. 1 – 22.

[15] Beightler C. S and Phillips D. T, "Applied Geometric Programming", Jhon Wiley & Sons, New York, 1976.

[16] Paul H and Tay, "Optimal Design of an Industrial Refrigeration System", In Proc. of Int. Conf. on Optimization Techniques and Applications, 1987, pp. 427 – 435.

DOI: 01.IJPIE.02.01.205